Computer Security Symposium 2019
21 - 24 October 2019

Webアプリケーションに対する攻撃検出・防御システムHoppinにおける攻撃検出方式の提案

武子 洸太郎[1,a)]　小出 洋[2,b)]

概要：Hoppin は既存の Web アプリケーションにハニーポットの機能を持たせることで攻撃の検出を行い，Web アプリケーションを攻撃から保護すると同時に攻撃手法の収集を行うことを可能にするシステムである．Hoppin は 404 エラー等が発生した場合にその接続を不正なものとみなし攻撃検出を行っていたが，本研究では新たに DNS を用いた攻撃検出方式を提案する．通常，Web アプリケーションの利用者は DNS による名前解決を行い Web アプリケーションに接続する．一方で攻撃者は，しばしば不特定のランダムな IP アドレスに対して攻撃を行うことがある．そのため，DNS コンテンツサーバのログに記録されていないクライアントからの接続は攻撃である可能性が高いと判断できる．そこで，本研究では Web アプリケーションへの名前解決を行う DNS コンテンツサーバを用意し，そのログを参照することで Web アプリケーションに対する接続者が名前解決を行っているか確認し，攻撃者であるかどうかを判断する．

キーワード：Web アプリケーション，ハニーポット ，攻撃検出，DNS

Proposal of Attack Detection Method on Hoppin, Attack Detection and Defense System for Web Applications

Kotaro Takeshi[1,a)]　Hiroshi Koide[2,b)]

Abstract: Hoppin is a system that makes it possible to detect attacks by giving honeypot functions to existing Web applications, protect Web applications from attacks, and collect attack methods. Hoppin considered that the connection was invalid and detected the attack when such as 404 error occurred. In this research, we propose a new method using DNS. Normally, users of web applications use DNS for name resolution and connect to web applications. On the other hand, attackers often attack unspecific IP addresses. Therefore, it can be judged that the connection from the client not recorded in the log of DNS content server is likely to be an attack. Thus, DNS content server that resolves the name to the Web application is prepared, and by referring to the log, it is confirmed whether the connection to the Web application is performing name resolution, and it is determined whether it is an attacker.

Keywords: Web Application, Honeypot, Attack Detection, DNS

1. はじめに

近年，サイバー攻撃の脅威は大きく増大している．情報通信研究機構（NICT）の調査では，サイバー攻撃関連通信の量が 2017 年は 2016 年から約 1.2 倍，2018 年は 2017 年から約 1.4 倍増加したと報告されている [1]．サイバー攻撃件数の増加のみならず，その手法の複雑化・高度化も大きな問題である．かつてのサイバー攻撃の多くは不

[1] 九州大学 大学院システム情報科学府 Graduate School and Faculty of Information Science and Electrical Engineering, Kyushu University Fukuoka 819-0395, Japan
[2] 九州大学情報基盤研究開発センター Research Institute for Information Technology Kyushu University, Fukuoka 819-0395, Japan
[a)] takeshi.koutarou.346@s.kyushu-u.ac.jp
[b)] koide@cc.kyushu-u.ac.jp

特定多数を標的としたものが多く，攻撃の発見も容易で速やかな対策が可能であった．しかし近年，Web アプリケーションの脆弱性を狙った高度な攻撃が増加しており，OWASP(Open Web Application Security Project) の調査では特にインジェクション等が重要な脆弱性であるとされている [2]．多くの場合これらは比較的目立たない攻撃であり，発覚が遅れるため被害が拡大しやすい傾向がある [3]．また，攻撃によって改竄された Web サイトに利用者がアクセスすることで悪性サイトへ誘導され，マルウェアに感染するといったケースも報告されている [4, 5]．そのため，それらの攻撃を検知して Web アプリケーションを防御しつつ攻撃手法を収集することが重要である．しかし，大規模かつ複雑な情報システムにおいて，既存の Web アプリケーションが持つ脆弱性を修正することは容易ではない．また，開発段階のみならず，運用段階でのセキュリティ対策が重要であると述べる先行研究も存在する [6]．ゆえに，運用中の Web アプリケーションに直接手を加えることなく，Web アプリケーションと密接に連携したセキュリティ機能を持つことが必要である [7]．

Web アプリケーションそのものに手を加えることなくセキュリティを向上する手段の一つとして Web Application Firewall(WAF) が挙げられる．WAF は Web アプリケーションが持つ脆弱性を悪用した攻撃を検出して，攻撃から Web アプリケーションを防御するためのセキュリティ機構である．WAF は Web アプリケーションへの通信内容を検査することによって，Web アプリケーションを XSS や SQL インジェクションから防御することができる．一方でなりすまし攻撃の場合，HTTP 通信自体は WAF にとって正常な通信として認識されてしまうため WAF で防御することができない．

また，サイバー攻撃の手法を収集するためのシステムとしてハニーポットが挙げられる．ハニーポットは意図的に脆弱性を持たせた囮を用意することで攻撃者の不正アクセスを誘発して，その攻撃手法を収集することを目的としたシステムである．ハニーポットには，大きく分けて高対話型ハニーポットと低対話型ハニーポットの二種類が存在する．高対話型ハニーポットは，実際のシステムに脆弱性を持たせて利用するものである．本物のシステムを利用するためより高度な情報を収集できるが，その分リスクが高く運用には細心の注意を払う必要がある．低対話型ハニーポットは，特定の環境をエミュレートすることで攻撃の監視を行う．本物のシステムを利用しないため比較的低いリスクで運用することが可能だが，その分高対話型と比べると収集できる情報量は少なくなる．

2. ハニーポット

2.1 概要

ハニーポットは意図的に脆弱性を持たせた囮を用意することで攻撃者の不正アクセスを誘発して，その攻撃手法を収集することを目的としたシステムである．ハニーポットはシステムの脆弱性を悪用する攻撃者から情報を収集する際に非常に有用である．Web ハニーポットへの攻撃の宛先 URL を変換することで，失敗した攻撃のうち約 50%から攻撃成功時と同程度の情報を収集できることを示した先行研究も存在する [8]．ハニーポットには，大きく分けて高対話型ハニーポットと低対話型ハニーポットの 2 種類が存在する．

2.2 高対話型ハニーポット

高対話型ハニーポットは，実際に脆弱性を持たせた OS やアプリケーションをハニーポットとして利用するものである．本物のシステムを利用するため，低対話型よりも高度な情報を多く収集できる．高対話型ハニーポットでは本物のシステムを利用しているため，物理的なマシンやファイルシステムの破壊，BOT として運用される等のリスクがある [9]．

2.3 低対話型ハニーポット

低対話型ハニーポットは，特定の OS やアプリケーションをエミュレートして，ハニーポットとして利用するものである．本物のシステムを利用せず，攻撃がホストマシンへ影響を及ぼすことがないため，高対話型と比べて低いリスクで運用することが可能である．一方で特定の攻撃手法に関する情報しか収集できないため，高対話型と比べると収集できる情報量は少なくなる．代表的な低対話型ハニーポットとして，SSH ハニーポットの Cowrie[10] や HTTP ハニーポットの Glastopf[11]，それらを統合した T-Pot[12] 等が存在する．

3. DNS[13, 14]

DNS（Domain Name System）は，ドメイン名と IP アドレスの対応関係を管理するシステムである．DNS にはドメイン名と IP アドレスの対応表を保持するコンテンツサーバと，クライアントからの名前解決依頼を受けてコンテンツサーバに対して問い合わせを行うキャッシュサーバが存在する．

3.1 DNS コンテンツサーバ

DNS コンテンツサーバは権威 DNS サーバとも呼ばれる．コンテンツサーバは図 1 に示すような木構造によって分散管理されている．各コンテンツサーバはそのコンテンツサーバが設置された階層のドメインに関する情報の他，より下位のドメインを管理するコンテンツサーバの情報等を管理する．

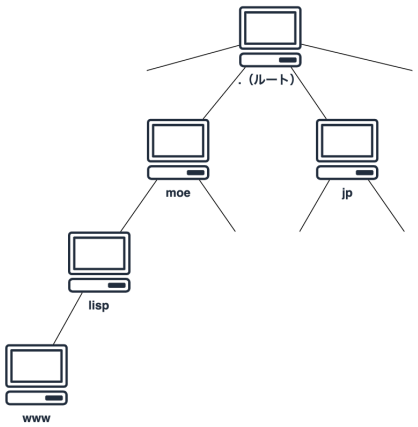

図 1　DNS コンテンツサーバの木構造
Fig. 1　Tree structure of DNS contents server

3.2　DNS キャッシュサーバ

DNS キャッシュサーバはクライアントから名前解決依頼を受けて，DNS コンテンツサーバを木構造の上位から順に辿っていくことで名前解決を行い，その結果をクライアントに返す．キャッシュサーバはコンテンツサーバの負荷軽減のために過去の問い合わせの結果を一定期間保存し，期間内に同じ問い合わせが来た場合にはキャッシュサーバ内に保存された情報をクライアントに返す．以下に「www.lisp.moe」に対する名前解決を例としてキャッシュサーバの動作を示す．

(1) クライアントがキャッシュサーバに IP アドレスを問い合わせる．
(2) キャッシュサーバはキャッシュに「www.lisp.moe」の IP アドレスが残っていた場合はクライアントにその IP アドレスを伝える．そうでなければルートサーバに「www.lisp.moe」の IP アドレスを問い合わせる．
(3) ルートサーバは「.moe」のドメインを管理する下位のコンテンツサーバの IP アドレスをキャッシュサーバに伝える．
(4) 同様の手順を繰り返し，最終的に「www.lisp.moe」のドメインを管理するコンテンツサーバがその IP アドレスをキャッシュサーバに伝える．
(5) キャッシュサーバがクライアントに「www.lisp.moe」の IP アドレスを伝える．

4.　攻撃検出システム Hoppin

4.1　概要

Honeypot in Web Application Framework(Hoppin) とは，Web アプリケーションへの攻撃を検出し，Web アプリケーションを攻撃から防御するとともに攻撃手法の収集を行うシステムである．Hoppin は HTTP リクエストを受け取ると，保存されたユーザ属性の値に応じて使用する API を正規用と擬似用に切り替えることで，Web アプリケーションを攻撃から防御すると同時に攻撃手法の収集を行うことが可能である．Hoppin の各システムとコンテンツは Docker[15] によって構築されている．Docker とは，Docker 社が開発しているコンテナ型の仮想空間を提供するソフトウェアである．コンテナ型の仮想空間は従来の仮想化技術とは異なり，OS を丸ごとインストールする必要がなく，ホスト OS 上のプロセスとして動作するため軽量であり，環境構築が容易である．Docker において各コンテナ内で実行されているプロセスは独立しており，他のコンテナに対して影響を与えることがない．Docker を用いたコンテナ技術は，ハニーポットのように攻撃者にシステム内への侵入を許すようなサーバーを運用するには最適である [9]．Hoppin は Docker を用いることで，高対話型ハニーポットと同等の機能を持ちつつ低対話型ハニーポットと同等の安全性を実現している [16]．

4.2　Hoppin の構成

Hoppin では，リバースプロキシ機能を有した攻撃検出システムと，収集した攻撃情報の記録・閲覧を行う管理システムの 2 つから構成されている．コンテンツへの外部からの接続は必ずリバースプロキシを通るため，コンテンツに外部から直接接続することは不可能である．コンテンツは 1 つでも動作するが，複数個のコンテンツを用意することで攻撃者に 1 つのコンテンツがハニーポットと判断された場合において，再接続の際に他のコンテンツに接続され，引き続き攻撃を誘発することができる．図 2 に Hoppin のシステム構成を示す．

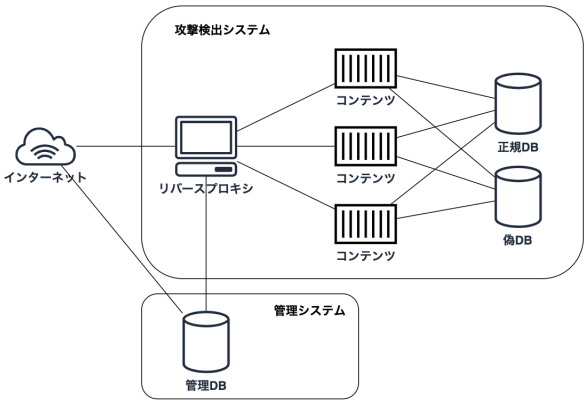

図 2　Hoppin のシステム構成
Fig. 2　System configuration of Hoppin

4.3　Hoppin の動作

ユーザが各コンテンツに接続を行う場合．攻撃検出システムを介して接続を行う．その際に実行される処理は以下の通りである．まず初回接続時に接続元の IP アドレスをキーとしてセッション ID を生成し，キャッシュを行う．次回以降の接続はキャッシュされた情報を読み出し，その

情報を用いて接続を行う．接続先情報が読み出されるとHoppinはHTTPリクエストの読み取りを行い，セッション情報を管理データベースに記録する．その後，Hoppinはユーザ属性の値に応じて情報の記録やHTTPリクエストの書き換えを行う．ユーザ属性の状態は通常モードと記録モードの2種類が存在する．通常モードは通常のユーザが接続を行う際に使用されるモードである．記録モードは，ユーザがHoppinに対して行った不正通信の回数が一定の値を超えた場合に移行するモードである．なお，モード切替の基準となる閾値は自由に変更することが可能であるため，誤検知率に応じて柔軟に対応することが可能である．記録モードに移行した後は，ユーザが接続を行うたびにHTTPリクエストの内容をデータベースに記録する．取得する情報はパスと接続メソッドの2種類である．接続メソッドがPOSTメソッドであった場合にはパラメータの情報も合わせて記録する．

また，APIに接続する場合には，コンテンツ側で用意された擬似用のAPIにパスの情報を書き換えた後にコンテンツにリクエストの送信を行う．これにより攻撃者は正規のAPIで接続を行なっていると錯覚させることが可能となり，更に高度な攻撃を誘発することが可能となる．コンテンツが使用するAPIやデータベースは，通常モード中に用いられる正規のものと記録モード中に用いられる偽のものの2種類が存在する．Hoppinはユーザ属性の値に応じてAPIを書き換え，使用するデータベースを変更する．以下に通常モードと記録モードにおける動作例をそれぞれ図3，図4に示す．

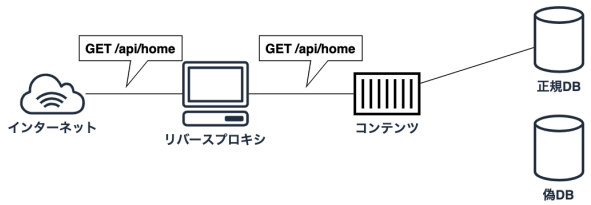

図 3 通常モードにおける動作
Fig. 3 Operation in normal mode

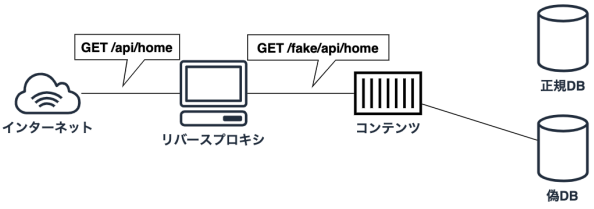

図 4 記録モードにおける動作
Fig. 4 Operation in recording mode

4.4 攻撃検出方式

Hoppinでは不正な通信を検出した際にモードスコアの値を増加させ，モードスコアの値が一定の閾値を超えた場合にその接続者を攻撃者であるとみなす．また，従来は一つの検出手法のみを用いていたが，攻撃検出精度を高めるために今回新たな検出手法を提案する．以下にそれぞれの手法を示す．

4.4.1 HTTP ステータスコードによる攻撃検出 (従来手法)

Hoppinは HTTP ステータスコード 401, 403, 404 を検出した際に，その通信を不正通信とみなす．これらのエラーは認証が必要なページにアクセスを試みた際や，存在しないファイルにアクセスを試みた際に発生するため，攻撃の疑いがあると判断される．

4.4.2 DNS による攻撃検出 (提案手法)

通常，Webアプリケーションの利用者はDNSによる名前解決を行いWebアプリケーションに接続する．一方で攻撃者は，しばしば不特定のランダムなIPアドレスに対して名前解決を行わずに直接攻撃を行うことがある．これまでHoppinではステータスコード404等のエラーを確認した際のみその通信を不正通信とみなしていたが，本研究ではHoppinにおけるDNSを用いた新たな攻撃検出方式を提案する．本提案ではドメインの取得とDNSコンテンツサーバの用意を行い，Webアプリケーションへの対応付けを行う．コンテンツサーバでWebアプリケーションへの名前解決を監視し，リバースプロキシでWebアプリケーションへの接続があった際にそのクライアントが直前にDNSによる名前解決を行なっていない場合に，そのクライアントが攻撃者である可能性が高いと判断する．システム構成を図5に示す．

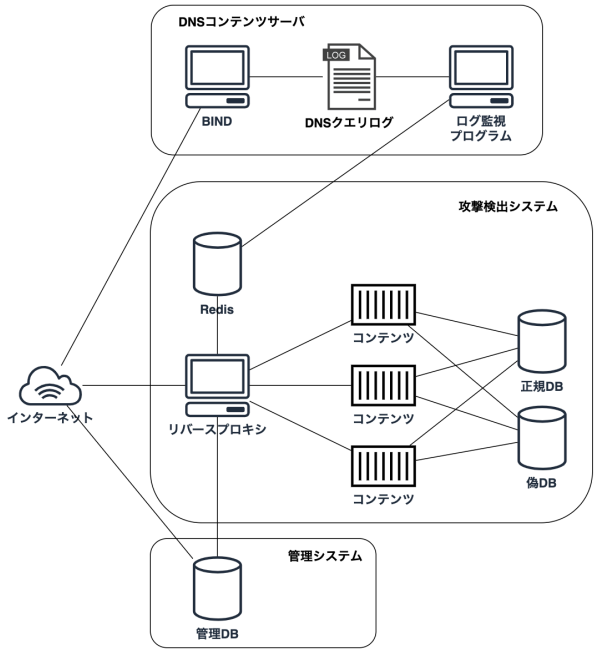

図 5 DNS を用いた攻撃検出システム
Fig. 5 Attack detection system using DNS

今回，DNS サーバとして BIND[13] を使用した．また，攻撃検出システム側で新たに Redis[17] を用意した．以下に提案するシステムの動作を示す．

(1) DNS コンテンツサーバ上で，クエリログを監視する．
(2) クエリログに更新があった際にそれが Web アプリケーションへの名前解決であれば，一定の値を攻撃検知システム上の Redis に登録する．
(3) その後リバースプロキシが Web アプリケーションへの接続を検出した際に，Redis にデータが登録されていれば正規の利用者であると判断する．

DNS コンテンツサーバのクエリログに記録される IP アドレスは接続者自身の IP アドレスではなく，接続者が使用している DNS キャッシュサーバの IP アドレスであるため，本提案では IP アドレスではなく名前解決と Web アプリケーションの接続のタイミングのみで正規の利用者であるかどうかの判断を行う．そのため，Redis に登録する値は IP アドレスに関わらず一定のものにした．また，DNS キャッシュサーバに Web アプリケーションの IP アドレスが記録されている間は，コンテンツサーバでの名前解決が行われず正規の利用者であるかどうかの判断が行えないため，今回の実装では DNS キャッシュサーバに Web アプリケーションの IP アドレスが記録される期間を非常に短く設定した．

4.5 今後実装予定の機能

今後の実装予定として既存の攻撃検出方式の改善や新たな検出方式の検討を行うほか，最終的には攻撃者毎にそれぞれ異なる仮想環境を Docker を用いて提供し，より発展的な攻撃手法を収集することを目標としている．図 6 に Hoppin の今後の構想を示す．

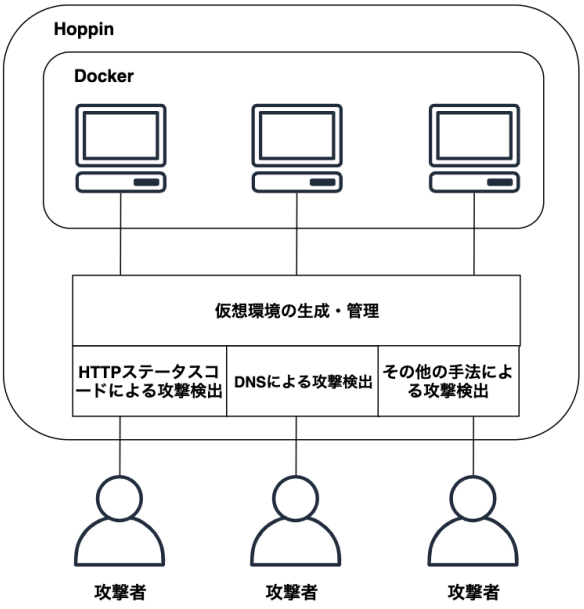

図 6 Hoppin の今後の構想
Fig. 6 Concept of Hoppin

5. まとめと今後の課題

本研究では，DNS を利用し接続者の名前解決の有無を確認することで Hoppin における新たな攻撃検出方式を提案した．しかし，今回の実装では DNS キャッシュの有効期間を非常に短く設定したため接続のたびに毎回名前解決が行われ，DNS コンテンツサーバ群に余計な負荷をかけるという問題点が存在する．そこで，DNS コンテンツサーバのクエリログに記録される接続者の DNS キャッシュサーバの IP アドレスと，リバースプロキシ側で検出する接続者自身の IP アドレスの紐付けを行う等の工夫を行い，DNS キャッシュの有効期限を長く設定し DNS コンテンツサーバ群に余計な負荷を与えずに，接続者の特定を行うことを可能とする仕組みの実装を今後の課題とする．

参考文献

[1] 国立研究開発法人情報通信研究機構: NICTER 観測レポート 2018 (オンライン), 入手先 ⟨http://www.nict.go.jp/cyber/report/NICTER_report_2018.pdf⟩ (参照 2019-08-21).
[2] OWASP: OWASP Top 10 - 2017 (online), available from ⟨https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf⟩ (accessed 2019-08-21).
[3] 総務省: サイバーセキュリティの現状と総務省の対応について (オンライン), 入手先 ⟨http://www.soumu.go.jp/main_content/000467154.pdf⟩ (参照 2019-08-21).

[4] 笠間 貴弘, 井上 大介, 衛藤 将史, 中里 純二, 中尾 康二:ドライブ・バイ・ダウンロード攻撃対策フレーム ワークの提案, コンピュータセキュリティシンポジウム 2011 論文集, pp.780-785 (オンライン) 入手先 ⟨https://ipsj.ixsq.nii.ac.jp/ej/⟩ (2011)
[5] 尾崎 幸也, 上山 真也, 小西 達也, 山崎 雅斗, 坂東 翼, 小林 孝史:悪性 URL の強調表示による Drive-by Download 攻撃解析支援手法の提案, コンピュータセキュリティシンポジウム 2017 論文集, pp.817-822 (オンライン) 入手先 ⟨https://ipsj.ixsq.nii.ac.jp/ej/⟩ (2017)
[6] 金 根學 , 原田 要之助: WEB アプリケーションのセキュリティマネジメントについての考察, 研究報告電子化知的財産・社会基盤（EIP）, pp.1-7 (オンライン) 入手先 ⟨https://ipsj.ixsq.nii.ac.jp/ej/⟩ (2017)
[7] T.Ishikawa and K.Sakurai:Parameter Manipulation Attack Prevention and Detection by Using Web Application Deception Proxy, Proc. IMCOM’17, pp.74:1-74:9 (2017)
[8] 八木 毅, 谷本 直人, 針生 剛男, 伊藤 光恭:高対話型 Web ハニーポットにおける攻撃情報収集方式の改善, コンピュータセキュリティシンポジウム 2009 論文集, pp.1-6 (オンライン) 入手先 ⟨https://ipsj.ixsq.nii.ac.jp/ej/⟩ (2009)
[9] 山本 健太, 齊藤 泰一:Linux コンテナ技術を利用した SSH ハニーポットの提案と評価, コンピュータセキュリティシンポジウム 2016 論文集, pp.770-776 入手先 ⟨https://ipsj.ixsq.nii.ac.jp/ej/⟩(2016)
[10] GitHub - cowrie/cowrie: Cowrie SSH/Telnet Honeypot (online), available from ⟨https://github.com/cowrie/cowrie⟩(accessed 2019-08-21).
[11] GitHub - mushorg/glastopf: Web Application Honeypot (online) urlehttps://github.com/mushorg/glastopf(accessed 2019-08-21).
[12] DTAG Community Honeypot Project: Release T-Pot 19.03 (online), available from ⟨http://dtag-dev-sec.github.io/mediator/feature/2019/04/01/tpot-1903.html⟩(accessed 2019-08-21).
[13] Internet Systems Consortium, Inc.: Internet Systems Consortium(online), available from ⟨https://www.isc.org/bind/⟩(accessed 2019-08-21).
[14] Mockapetris, P.: DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION(RFC1035), IETF(online) available from ⟨https://www.ietf.org/rfc/rfc1035.txt⟩(accessed 2019-08-21).
[15] Docker Inc: Enterprise Container Platform — Docker (online), available from ⟨https://www.docker.com/⟩(accessed 2019-08-21).
[16] 野見山 賢人, 小出 洋:Web アプリケーションのための攻撃検出と防御, コンピュータセキュリティシンポジウム 2017 論文集, pp.172-177 入手先 ⟨https://ipsj.ixsq.nii.ac.jp/ej/⟩(2017)
[17] Redis Labs: Redis Labs — Database for the Instant Experience (online), available from ⟨https://redislabs.com/⟩(accessed 2019-08-21).